Analogical Model Formulation for AP Physics Problems

Matthew Klenk, Kenneth D. Forbus

Qualitative Reasoning Group, Northwestern University
2133 Sheridan Road, Evanston, IL, 60208
{m-klenk, forbus}@northwestern.edu

Abstract

One of the least studied problems in qualitative reasoning is computing structural descriptions, i.e., how to move from the unruly, broad set of concepts used in everyday life to a concise, formal vocabulary of abstractions that can be used effectively for problem solving. This paper describes how learning by analogy can be used to solve this problem in the domain of AP physics problems. The system starts with some basic mathematical skills, a broad ontology covering many aspects of everyday life, and some basic qualitative mechanics. By studying worked solutions, it learns equations and modeling strategies that can be used to solve AP Physics problems. By examining systematic variations of problems, we show that analogical model formulation can be used to solve three kinds of transfer problems: parameterization, restructuring, and restyling.

Introduction

One of the important contributions of qualitative reasoning has been formalizing the process of model formulation (cf. Falkenhainer & Forbus, 1991; Nayak 1994, Rickel & Porter, 1994). Most model formulation work has focused on ascertaining what levels of detail and which perspectives should be used in a model, given a particular task. These algorithms take as input a structural description of the system to be modeled, a kind of high-level schematic. On the whole, they do very little reasoning about the structural descriptions. An exception is analog electronics, where (Flores & Cerda, 2000) shows how to formalize a number of equivalent circuit configurations as rewrite rules, to simplify circuit schematics in a human-like way to make analyses more tractable. But otherwise, the problem of constructing such structural descriptions from everyday inputs has received little attention.

One class of problems where this issue arises is solving textbook physics problems. A major challenge for students is learning how to reframe familiar everyday situations into models that can be solved using the formal techniques of physics. This is challenging for a variety of reasons. The conditions under which particular equations are applicable must be learned. How particular conditions translate into parameter values must be learned, e.g., an object at rest has zero velocity, and objects on Earth are affected by gravity. A coin being dropped from a building might be approximated as a point mass, but the same coin being spun on a table (in a problem about angular momentum) cannot be viewed so. Learning to construct formal models that can be analyzed mathematically from everyday situations is one of the essential skills of a good scientist, so the importance of these skills goes well beyond physics itself.

Our hypothesis is that people learn how to formulate models via analogy. That is, they use their experience (both everyday and with solving textbook problems) to enable them to solve new problems, and over time, extract generalizations and heuristics. These enable them to perform well under a broad variety of circumstances. This is consistent with Falkenhainer's (1992) observation that engineers often use analogies with their experience to formulate new models, but goes beyond it, in focusing on constructing structural descriptions as well as learning aspects of the domain itself. In (Klenk *et al* 2005) we showed that modeling assumptions could be learned by analogy to solve everyday reasoning problems, of the kind found on the Bennett Mechanical Comprehension test. In this paper we go beyond that result, by looking at whether or not such techniques can be used to learn to solve Advanced Placement (AP) physics problems. The AP tests are tests taken by high school graduates to pass out of college level courses.

With many AI ideas and systems focused on broad the concept of learning, it is important to explicitly differentiate between different types of learning to guide further research in the field. One important distinction to consider is the amount of transfer from problems in a training set to problems in a test set. This characterizes how general what is learned is, i.e., how broadly can it be reused? In this paper we explore three distinct near-transfer problems[1]:

[1] These levels are from a 10-level catalog of transfer tasks used in DARPA's Transfer Learning Program (http://fs1.fbo.gov/EPSData/ODA/Synopses/4965/BAA05-

- *Parameterization*: Can problems that vary in terms of the specific numerical parameters be solved?
- *Restructuring*: Can problems with the same qualitative structure, but involving a different solution variable, be solved?
- *Restyling*: Can problems with the same qualitative structure, but different types of everyday objects, be solved?

All three types of transfer are important. It is hard to say that a system learned anything if it cannot solve the kind of trivial variation involved in parameterization. The ability to solve problems involving restructuring provides evidence that the system is learning in a way that goes beyond slavishly following the same procedure each time. Restyling addresses the need to go beyond the specific circumstances in which a principle was learned, to a broad set of situations in which it is applicable.

This paper describes an analogical learning system which exhibits the ability to perform these three kinds of transfer on a set of AP physics problems. We start by reviewing the analogical processing techniques used, then describe how the solver works, illustrating its operation with a particular problem. Then we discuss the results of an experiment showing that it is capable of performing these three types of transfer. We close with related work not mentioned elsewhere in the paper, and plans for future work.

Background

We use Gentner's structure-mapping theory of analogy and similarity (Gentner, 1983). In structure-mapping, analogy and similarity are defined in terms of structural alignment processes operating over structured representations. The output of this comparison process is one or more mappings, constituting a construal of how the two entities, situations, or concepts (called base and target) can be aligned. A mapping consists of a set of correspondences, a set of candidate inferences, and a structural evaluation score. A correspondence maps an item (entity or expression) from the base to an item in the target. A candidate inference is the surmise that a statement in the base might hold in the target, based on the correspondences. The structural evaluation score indicates overall match quality.

We use two cognitive simulations based on structure-mapping theory here. The *Structure-Mapping Engine* (SME) does analogical mapping (Falkenhainer et al, 1986). SME uses a greedy algorithm to compute approximately optimal mappings in polynomial time. The base and target descriptions can be pre-stored cases, or dynamically computed based on queries to a large knowledge base (Mostek et. al, 2000). MAC/FAC (Forbus et al, 1994) models similarity-based retrieval. The first stage uses a special kind of feature vector, automatically computed from structural descriptions, to rapidly select a few (typically three) candidates from a large case library. The second stage uses SME to compare these candidates to the probe description, returning one candidate (or more, if they are very close) as what the probe reminded it of. As performance systems, both SME and MAC/FAC have been used successfully in a variety of different domains, and as cognitive models, both have been used to account for a variety of psychological results (Forbus, 2001). Now we show how these domain-independent simulations can be used to solve physics problems from worked solutions.

Solving physics problems by worked solutions

When students study for the AP Physics exam, one important way in which they learn is by doing problem sets. The feedback students often get is in the form of worked solutions in the back of the book. We believe that this is a promising type of learning to explore. Through collaboration with the Educational Testing Service (ETS), the organization that administers the AP Physics exam, we obtained a number of example problems, illustrating a variety of types of problems found on the exam and worked solutions for each problem. We initially are focusing on Newtonian dynamics. We translated the problems and the worked solutions into predicate calculus, using the contents drawn from the ResearchCyc knowledge base plus our own extensions. ResearchCyc is useful for this purpose because it includes over 30,000 distinct types of entities, over 8,000 relationships and functions, and 1.2 million facts constraining them. Thus everyday concepts like “astronaut” and “ball” are already defined for us, rather than us generating them specifically for the purpose of this project.

Example Worked Solution

Consider the following physics problem:

An astronaut on a planet with no atmosphere throws a ball upward from near ground level with an initial speed of 4.0 m/s. If the ball rises to a maximum height of 5.0 m, what is the acceleration due to gravity on this planet?

Our formal version of the ETS-supplied worked solution for this problem has the following steps:

1. Recognize and instantiate the distance-velocity under constant acceleration equation for the ball’s motion (Vi^2 = Vf^2 – 2ad)
2. Given the projectile motion event infer that the velocity of the ball at the maximum height is 0 (Vf = 0 m/s)

3. Given the projectile motion and the lack of atmosphere, infer that the value of the acceleration on the ball is the gravitational force of the planet (a = g)
4. Given that the ball is thrown from near the ground, assume the height of the ball is equal to the distance the ball travels during the upward motion (d = h, given as 5.0 meters)
5. Apply the previous steps to solve the distance-velocity equation for acceleration (g = 1.6 m/s/s)

```
(solutionStepUses Step4-P2-WS
  (isa Ground-2 SurfaceRegion-Tangible))
(solutionStepUses Step4-ETS-P2-WS
  (groundOf HYP-Planet-2 Ground-2))
(solutionStepUses Step4-ETS-P2-WS
  (eventOccursNear ThrowingEvent-2 Ground-2))
(solutionStepResult Step4-ETS-P2-WS
  (math=
     (DistanceTraveled Ball-2
      (TimeFn (StartFn ProjectileMotion-2)
             TimePoint-2))
     (Height Ball-2 HYP-Planet-2
            TimePoint-2)))
(solutionStepOperation Step4-ETS-P2-WS
                        AssumedEquation)
```

Figure 1: Step 4 of Problem 2 worked solution

Figure 1 shows the formal description of Step 4, for concreteness. Notice that the description of the worked solution is in terms of high-level operations, not the internal reasoning vocabulary of our problem solver. This is important, since this provides more opportunities for the system to learn (and to make mistakes). We store the worked solution along with the problem description as a case in our case library, which will be used to solve new problems. This is a very simple form of learning, learning by accumulating examples. We discuss our plans to move beyond this below.

Solving a Problem

Problem instances are presented as cases containing a statement indicating the goal parameter. The first phase of problem solving is to generate a mapping with a relevant example. This is done in three steps. First, the problem solver retrieves an example from the case library using the problem case as the probe in MAC/FAC. Second, it creates an *orienting mapping* between the relevant example and the probe in which only statements concerning the qualitative event structure are considered. Figure 2 shows the representation of event structure for our running example. It includes the events that occur in the description of the problem, the statements about the entities that participate in those events and relationships between the events. Since dynamics is about the properties of objects undergoing particular kinds of events, ensuring that the qualitative event structure is accurately aligned provides a solid basis for importing knowledge from a worked example into a new problem.

Next, the retrieved example and the problem are compared, but using the correspondences found in the orienting mapping as constraints on this new mapping. The candidate inferences of this new mapping will include the solution steps for the worked problem. These solution steps are used as necessary in the general problem-solving process. But before any step can be used, it is inspected to ensure that it is applicable in the current problem.

```
(isa Astronaut-2 Astronaut)
(isa Ball-2 Ball)
(isa HYP-Planet-2 Planet)
(isa ProjectileMotion-2 ProjectileMotion)
(eventOccursAt ProjectileMotion-2
              HYP-Planet-2)
 (primaryObjectMoving ProjectileMotion-2
                     Ball-2)
(eventOccursNear ThrowingEvent-2 Ground-2)
(isa ThrowingEvent-2 ThrowingEvent)
(causes-EventEvent ThrowingEvent-2
                  ProjectileMotion-2)
(contiguousAfter ProjectileMotion-2
                ThrowingEvent-2)
(performedBy ThrowingEvent-2 Astronaut-2)
```

Figure 2: Qualitative event structure for Problem 2

The general problem-solving process concerns finding the value for a quantity. The system can find the value of a quantity in three different ways. First, it may already be known as part of the problem. Second, it may be able to find an applicable solution step from the candidate inferences in the mapping in which the goal parameter is assumed. Third, an applicable solution step indicating a relevant equation containing the sought after quantity. With this equation, the system can recursively search for values of the other quantities in the equation until the equation is solvable for the sought after quantity. It is important to note that the current version of the system does not start with any of the equations of physics in a general form – it only has access to them through examples of how they have been used, in the worked examples. Thus analogical reasoning is essential to the system being able to solve any physics problems.

To determine whether or not a solution step suggested by candidate inferences is valid for a given problem, the system checks the context surrounding the previous use of the solution step. Suppose for example the step assumes that the only force on a dropped ball is the force of gravity, because there is no atmosphere on the planet in the worked solution. There has to be a corresponding fact saying there is no atmosphere on the planet in the problem, or the ability to infer that there is no atmosphere, if this step is to be applied. These context facts act as preconditions that must be verified for the inferred step to be usable.

The algebra routines are simple, currently based on the system in (Forbus & de Kleer 1993). We currently treat the mathematical operations involved in solving a problem as a black box, not subject to learning.

Modeling knowledge in worked solutions

As evident from the worked solution steps, problem solvers are required to make a variety of modeling assumptions to successfully find the solution. First, the problem solver must determine which equations are applicable for a given situation. This is required because, even in a relatively constrained domain such as physics, the number of equations mentioning common variables such as acceleration is quite large. Efficient problem solvers should not exhaustively search this space. Second, a problem solver has to make assumptions to infer parameter values. In the example above, the problem is not solvable if the problem solver fails to recognize that the ball has zero velocity at its highest point. Third, there is the notion of default circumstances. The most common of these in AP physics is to assume that events happen on Earth and are subject to Earth's gravity unless otherwise mentioned. Finally, simplifying assumptions, such as viewing an object as a point mass or assuming a collision is elastic, are often required to make complex situations tractable.

Three of the four types of modeling assumptions are handled by our system directly through analogical reasoning. That is, determining parameter values, default circumstances, and relevant equations are handled directly by the analogy with the worked solution. Only the last type, categorizing an everyday object in terms of an abstraction, is not currently handled by our system. Instead, we take the categorization as acceptable if it is compatible with the rest of the mapping. This works well when the analogous problems are close, but could run into trouble when the analogs are more distant.

Learning conditions for such categorizations is one of our goals, but it turns out to be complex. Worked solutions for people provide at best partial information about why a modeling assumption they used is reasonable. For example, modeling the ball as a point mass in the example problem is never justified on other grounds. Students are expected to generalize from a body of examples they have seen about when to apply such ideas, probably in part because the ontology of everyday things is so broad, and the subset of object types that are appropriate for a particular idealization are not tightly localized to one part of the ontology. For example, rocks, coins, soda cans, and ferrets can all be considered as point masses for some kinds of problems, but most ontologies would not consider these categories as being particularly close otherwise.

An Experiment

We conducted an experiment to investigate this model. We chose three types of problems, of the kind found on AP physics tests, provided to us by the Educational Testing Service. Examples of these problem types are:

1. A ball is released from rest from the top of a 200 m tall building on Earth and falls to the ground. If air resistance is negligible, which of the following is most nearly equal to the distance the ball falls during the first 4 s after it is released?
2. An astronaut on a planet with no atmosphere throws a ball upward from near ground level with an initial speed of 4.0 m/s. If the ball rises to a maximum height of 5.0 m, what is the acceleration due to gravity on this planet?
3. A 5.0 kg object is moving with speed v when it makes a head-on collision with a 2.0 kg object, initially at rest. If friction is negligible, what must be the speed v, if after the collision the 5.0 kg object has speed 1.0 m/s, the 2 kg object has speed 2.5 m/s, and both objects are moving in the same direction?

To test the verification of modeling assumptions, a copy of each problem case was made in which the cases were missing a precondition for a necessary modeling assumption. We call these *bogus* problems, since they look like they should be analogous but are not. An example would be that problem type 3 states that you can ignore friction on both blocks. Without this given information, the problem becomes unsolvable because one cannot instantiate the conservation of momentum equation.

Each original problem was represented as a case stored with its worked solution in the case library. Then copies of each problem, but not the worked solution, were created for each of the transfer conditions – parameterization, restructuring, and restyling, as per above – along with a bogus problem. For parameterization problems, the values for all the parameters mentioned in the problem statement were changed, but the objects included all remained of the same types as before. For example, the parameterization of problem 2 involved a ball thrown upwards at 1 m/s and it reached a height of 10 meters. For restructuring problems, the system was asked to solve for a different quantity. For example, the restructuring of problem 3 provided the value of the initial velocity of the initially moving object as 2.5 m/s and asked for the velocity of that object post collision.

For restyling, the entity types were all changed but the event types remained the same. For example, the restyling of problem 1 included dropping a block off of a ledge.

The four versions of each problem were given to the system. Its results are summarized in Table 1. We scored a problem as being correct for the parameterization, restructuring, and restyling conditions if the system produced the correct answer. Because bogus problems do not provide enough information to be solved correctly, they were scored as correct when the system could not produce an answer because it could not prove a precondition for an assumption it attempted to make while solving the problem.

	Problem 1	Problem 2	Problem 3
Parameterization	Correct	Correct	Correct
Restructuring	Correct	Correct	Correct
Restyling	Correct	Correct	Failed
Bogus Problem	Correct	Correct	Correct

Table 1: Experimental Results

The system's only failure was Problem 3's restyling problem. Examination of the system's explanations for its results revealed that the error was due to a mismatch in the orienting mapping. The motion events of each object after collision are extremely symmetric, making it equally likely that SME would choose to match the motion of object-1 after the collision in the new problem with the motion of object-2 as with the motion of object-1 in the worked solution. We discuss remedies for this problem below.

Related Work

Physics problem solving is a classic domain for qualitative reasoning, starting with de Kleer's (1977) pioneering work in reasoning about sliding motion problems. Subsequent work mostly focused on equation-solving, and we used the results from Bundy (1983) in designing our equation-solver. Pisan (1996) exploited qualitative representations to reason about modeling assumptions in engineering thermodynamics, but did not explore learning issues.

AI research on analogy in problem solving has a similarly long history, including (Carbonell, 1986; Melis & Whittle, 1999; Veloso and Carbonell 1993). The closest systems to ours are Cascade (VanLehn & Jones, 1993; VanLehn 1998) and APSS (Ouyang and Forbus, 2006). Cascade starts with equations and other domain knowledge, and only attempts to learn search control heuristics. By contrast, our search control mechanism is currently fixed. APSS is built on Pissan's TPS system, and uses analogy to solve textbook engineering thermodynamics problems. Like TPS, it starts with a full suite of domain knowledge and can solve problems without prior examples purely from first principles, with analogy serving only to improve performance. We differ from both Cascade and APSS in that we focus on learning domain knowledge.

Discussion

This paper has examined how a learner can go from the unruly, broad common sense world to the refined world of parameters, equations, and modeling assumptions. While the overall performance of the system is already quite good, so far, we are have only tested it on three types of problems out of more than 20 which are relevant to the domain. Our future work is motivated by the goal of expanding the system to the point where it can learn all of the material on an AP physics exam about Newtonian dynamics.

In addition to testing the system on more problems and problem types, there are certain additions that we believe to be essential to handle a larger exam involving more complex transfer. First, we need to incorporate *rerepresentation* (Yan *et al* 2003) to overcome errors in analogical matching. The one failure we had can, as described above, be traced back to a mapping error, and this particular error is already handled by the existing theory, although the implementation did not use it. Second, AP physics problems often include diagrams, as do worked solutions. We plan to incorporate sKEA (Forbus & Usher, 2002) to enable ETS to create sketches for problems and worked solutions, and extend our system to be able to exploit sketches in its reasoning. For example, knowing that the direction of motion is downward when something falls is an example of a piece of common sense that we should be able to automatically extract from sketches in worked solutions. Third, we plan to move beyond learning by accumulating examples, in several ways. We plan to construct generalizations based on SEQL (Kuehne *et al* 2000) to facilitate the system's ability to transfer what it learns more broadly. For example, equations might be learned as *encapsulated histories* (Forbus, 1984), which, being parameterized, could extend the system's reach still further. Fourth, we plan to use analogical generalization over a corpus of physics problems to learn category assignments of everyday categories to structural abstractions. As Chi et al. (1981) note, one difference between novices and experts appears to be in their encoding strategies: Novices sort problems according to the kinds of objects that appear in them, while experts sort them according to the principles they would use to solve them. Consequently, we plan to explore methods for learning new encoding strategies, to capture this ability to move more directly from the everyday world to models that can be used to solve problems.

Acknowledgments

This research was supported by the Information Processing Technology Office of the Defense Advanced Research Projects agency. The opinions expressed herein are solely those of the authors, and do not represent the US government. We thank Pat Kyllonen and the Educational Testing Service for providing sample problems.

References

Bundy, A. 1983. *The Computer Modeling of Mathematical Reasoning*. Academic Press.

Carbonell, J. 1986. Derivational analogy: A theory of reconstructive problem solving and expertise acquisition. In R. Michalski, J. Carbonell, and T. Mitchell (Eds.) *Machine Learning: An artificial Intelligence Approach.* pp 137-161, Springer.

de Kleer, J. 1977. Multiple representations of knowledge in a mechanics problem solver, pp.299–304. *Proc. IJCAI-77.*

Falkenhainer, B. 1992. Modeling without amnesia: Making experience-sanctioned approximations. *Proceedings of QR02.*

Falkenhainer, B. and Forbus, K. 1991. Compositional modeling: finding the right model for the job. *Artificial Intelligence* 51:95–143.

Flores, J. and Cerda, J. 2000. Efficient modeling of linear circuits to perform qualitative reasoning tasks. *AI Communiations*, **13**(2) 125-134.

Forbus, K. 1984. Qualitative process theory. *Artificial Intelligence* 24:85–168.

Forbus, K. Exploring analogy in the large. In Gentner, D., Holyoak, K., and Kokinov, B. (Eds.) *Analogy: Perspectives from Cognitive Science.* MIT Press. 2001.

Forbus, K. and de Kleer, J., *Building Problem Solvers*, MIT Press, 1993.

Forbus, K., Gentner, D., and Law, K. MAC/FAC: A model of similarity-based retrieval. *Cognitive Science*, 19, 141-205. 1994.

Forbus, K. and Usher, J. 2002. Sketching for knowledge capture: A progress report. *Proceedings of IUI'02*, January 13-16, 2002, San Francisco, California.

Gentner, D., Structure-mapping: A theoretical framework for analogy, *Cognitive Science* 7(2), 1983.

Kuehne, S., Forbus, K., Gentner, D., and Quinn, B. 2000. SEQL: Category learning as incremental abstraction using structure mapping. *Proceedings of CogSci-2000.* Philadelphia, PA.

Melis, E. and Whittle, J. 1999. Analogy in inductive theorem proving. *Journal of Automated Reasoning*, 22:117-147. Kluwer.

Mostek, T., Forbus, K, Meverden, C. Dynamic case creation and expansion for analogical reasoning. *Proceedings of AAAI-2000.* Austin, TX. 2000

Nayak, P. 1994. Causal approximations. *Artificial Intelligence* 70:277–334.

Ouyang, T. and Forbus, K. 2006. Strategy variations in analogical problem solving. *To appear in Proceedings of AAAI-06*

Rickel, J. and Porter, B. 1994. Automated modeling for answering prediction questions: selecting the time scale and system boundary, pp. 1191–1198. *Proc. AAAI-94.*

VanLehn, K. 1998. Analogy Events: How Examples are Used During Problem Solving. *Cognitive Science* 22(19), pp 347-388.

VanLehn, K. and Jones, R. 1993. Better learners use analogical problem solving sparingly. In *Proceedings of the 10th International Conference on Machine Learning*, pp 338-345, Morgan-Kauffman.

Veloso, M. and Carbonell, J. 1993. Derivational analogy in PRODIGY: Automating case acquisition, storage, and utilization. *Machine Learning*, 10:249-278.

Yan, J., Forbus, K. and Gentner, D. 2003. A theory of rerepresentation in analogical matching. *Proceedings of the 25th Annual Conference of the Cognitive Science Society*